EVOLUTIONARY ROBOTS: THE NEXT GENERATION

Dario Floreano and Joseba Urzelai
Laboratory of Microprocessors and Interfaces (LAMI)
Swiss Federal Institute of Technology (EPFL)
CH-1015 Lausanne, Switzerland

February 28, 2000

Abstract

After reviewing current approaches in Evolutionary Robotics, we point to directions of research that are likely to bring interesting results in the future. We then address two crucial aspects for future developments of Evolutionary Robotics: choice of fitness functions and scalability to real-world situations. In the first case we suggest a framework to describe fitness functions, choose them according to the situation constraints, and compare available experiments in the literature on evolutionary robotics. In the second case, we suggest a way to make experimental results applicable to real-world situations by evolving online continuous adaptive controllers. We also give an overview of recent experimental results showing that the suggested approaches produce qualitatively superior abilities, scale up to more complex architectures, smoothly transfer from simulations to real robots and across different robotic platforms, and autonomously adapt in few seconds to several sources of strong variability that were not included during the evolutionary run.

1 Introduction

Evolutionary Robotics is the application of artificial evolution to robotic systems with a sensory-motor interface to the world. Although in the early days artificial evolution was mainly seen as a strategy to develop more complex and performant robot controllers, nowadays the field has become much more sophisticated and diversified. We can identify at least three approaches to Evolutionary Robotics: Automated Engineering, Artificial Life, and Synthetic Biology/Psychology. These three approaches largely overlap with each other, but still have quite different goals that eventually show up in the results obtained.

Automated Engineering is about the application of artificial evolution for automatically developing algorithms and machines displaying complex abilities that are hard to program with conventional techniques. Within this context the desired architectures are well defined and the problem is usually cast in terms of parameter optimization by evolutionary techniques. Artificial evolution can come up with strikingly efficient and surprising solutions that exploit *invariants* and features invisible to an external observer[1]. For example, in one of our early experiments [30] we evolved

[1]Invariants are constant relationships. Our visual system exploits many invariants. For example, the fact that we perceive objects always the same size irrespective of distance is given by neural detectors that

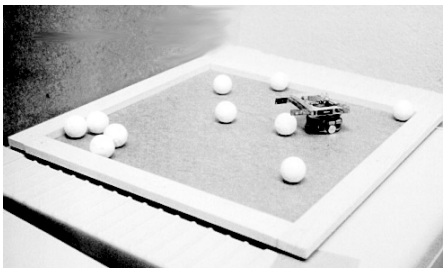

Figure 1: A Khepera robot evolved to find and pick up balls [30].

the control system of a Khepera robot with a gripper for the ability to find the highest number of ping-pong balls scattered around an arena and pick them up (figure 1). This was a difficult task because the resolution of the infrared sensors in front of the robot was not sufficiently good to discriminate between balls and walls. The best evolved robots succeeded using an unexpected strategy. They moved backwards until they detected something and then started to rotate on the spot until they faced the object, picking it up if it was a ball or resuming backward motion otherwise. This was possible because the rotation generated a scan of the object across several neighboring sensors whose combined activations were sufficient to discriminate between objects and assume a correct gripping position if necessary. At the same time, the remarkably simple avoidance strategy, coupled with the geometry of the arena, ensured a good exploration of the full environment.

Artificial Life instead is about evolution of artificial creatures that display life-like properties. In this context the notion of evolutionary goal is not appropriate (living creatures do not evolve towards a prespecified goal) and there are very few constraints that limit the directions that evolution might take. These evolutionary systems are usually self-sufficient, self-contained, and autonomous. The selection criterion is often the energy level of the creature and the environment has an ecological validity in that it includes food sources, mates, predators, a nest, etc. These artificial worlds are easier to implement in computer simulations because simulations give more freedom to experiment life-as-it-could-be. In this context, even evolutionary experiments that end up in complete extinction of one species, or display alternating dynamics such as in competitive co-evolutionary scenarios [35, 5, 12] (figure 2), may be considered important because they reveal interesting patterns of life. The artificial life approach is more interested in the emergent phenomena than in the optimization of a pre-defined strategy.

Synthetic Biology/Psychology attempt to understand the functioning of biological and psychological mechanisms by evolving those mechanisms in a robot put in conditions similar to those of the animals under study. This approach finds its roots in the inspiring booklet *Vehicles* [4] by the neurophysiologist Valentino Braitenberg who showed that apparently complex behaviors and emotions can be reproduced in the eye of an external observer by simple sensory-motor machines.[2] The evolutionary approach expands this method to more complex mechanisms and environmental conditions. For example, it is commonly assumed that rats use geometric modules

exploit the constant relationship between the size of the retinal projection and the perceived distance of the object. Evolved robots often discover invariants and exploit them to accomplish a task. Notice that since invariants exist mainly from the perspective of the subject perceiving them [18], it is virtually impossible for an external observer to define and incorporate them in a pre-fabricated control software.

[2]Braitenberg was also the first person to suggest artificial evolution of robots in that same booklet.

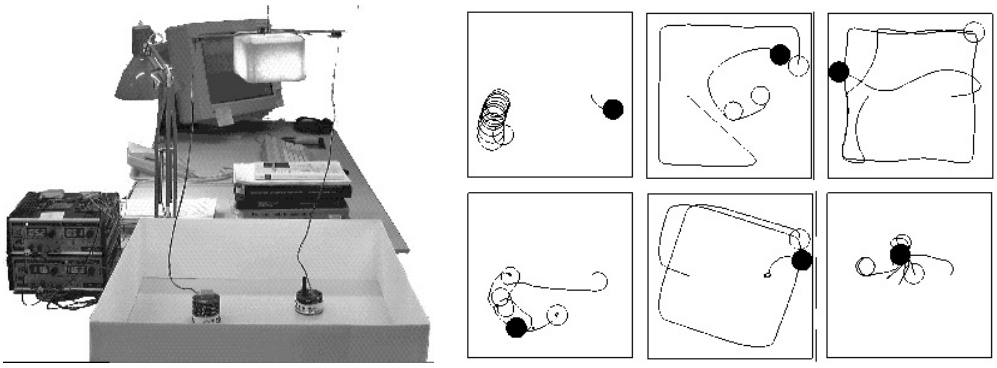

Figure 2: Co-evolutionary predator and prey robots display rapid alternation of diversified strategies caused by continuously changing dynamics. Left: co-evolutionary arena where a predator with vision is co-evolved with a short-sighted but faster prey. Right: Examples of emerging chasing-escape strategies every 10 generations (predator is black disk, prey is empty disk) [14].

and complex cognitive manipulations to find their way to a learned location [17], but behavioral and neurophysiological data are open to different interpretations and are still subject of heated scientific discussion. Orazio Miglino and Henrik Lund [26] investigated this issue by evolving Khepera robots in the same environmental conditions used for rats by ethologists and neurophysiologists. Their evolved robots reported the same performances measured on living animals, but did not use complex cognitive abilities to do so. Instead, these robots displayed simple sensory-motor sequences that capitalized upon geometric invariants of the environment. For example, the layout of long and short walls, combined with certain rotations by the robot, returned a high-probability to end up in the target location, no matter where the robot was initially located. Of course it would be wrong to deduce that rats use similar sensory-motor mechanisms, but evolutionary robotic data show that there is at least one *alternative explanation* for the available biological evidence. Therefore, the power of this approach is that it can be used to debunk strong assumptions based on weak evidence and at the same time suggest additional experiments.

These three approaches generate awareness and draw attention to different aspects of autonomous systems. The overall emerging picture is one where what matters most and differentiates Evolutionary Robotics from other machine learning approaches is that here robots self-organize while freely interacting with their own environment.

2 Research Areas

Evolutionary Robotics build upon several aspects of artificial evolution, as shown in figure 3. We believe that some of them are more promising than other. One way to read figure 3 is to visually organize it in three rows.

The bottom row includes aspects that typically pertain to machine learning and computer science. They include the best way to describe the searching properties of an evolutionary algorithm, how to design and optimally combine evolutionary operators, and how to implement the algorithms in software and hardware so that computational and physical resources are optimally exploited. These aspects are very important for conventional function optimization, but we argue that they are rather secondary for

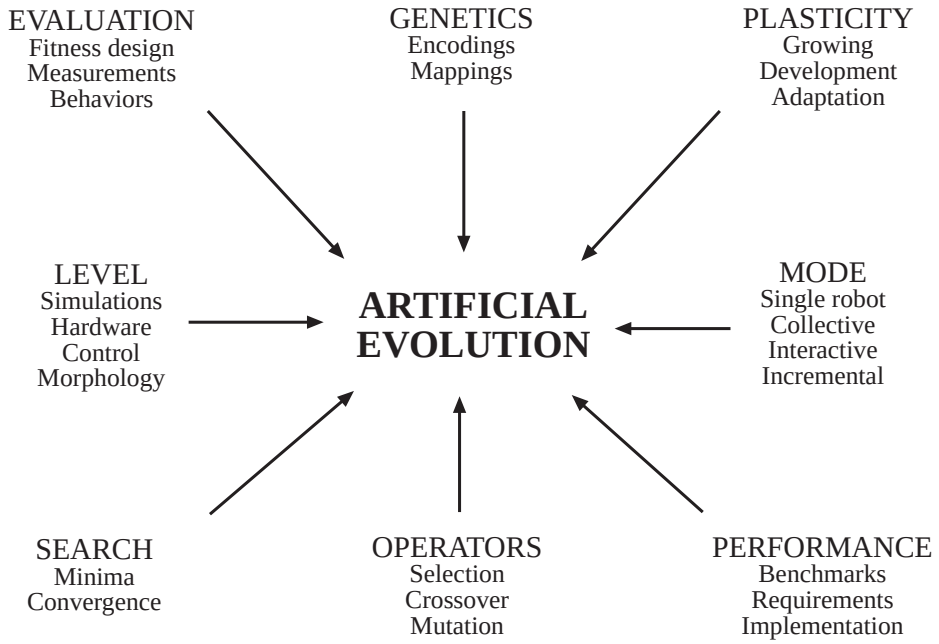

Figure 3: Different aspects of research in Artificial Evolution. The bottom row concerns mainly issues related to conventional evolutionary computation. The middle row displays hot issues that are currently being investigated mainly in the area of Evolutionary Robotics. The top row describes areas that have been only scarcely tackled and represent high potentials for significantly new achievements.

what concerns evolution of autonomous intelligent robots.

Machine Learning	**Robot Learning**
Learning *in vacuum*	Embedded Learning
Statistically well-behaved data	Data distribution not homogeneous
Mostly off-line	Mostly on-line
Informative feedback	Qualitative and sparse feedback
Computational time not an issue	Time is crucial
Hardware does not matter	Hardware is a priority
Convergence proof	Empirical proof

Table 1: Some crucial differences between Machine Learning and Learning in Autonomous Robots.

To start with, machine learning and robot learning are quite different, as shown in table 2. From an algorithmic perspective, an evolving robot spends by large most of its time interacting with the environment. Genetic operators and operations that map sensors into motor commands may take less than 5% of the total time. The remaining 95% is taken by mechanical actions, such as move a leg, rotate the camera, update the visual field, transmit signals across various parts of the hardware (and/or to an external workstation), etc.

An experiment could miserably fail if one does not pay sufficient attention to the interaction aspects even if the searching properties and computational efficiency of the evolutionary engine were optimally set. For example, if the robot is started always from the same position, the fitness is quickly maximized but the results may not generalize to different starting positions. On the other hand, by only taking a quick look at the experimental literature it turns out that almost any type of selection strategy (ranking-based, proportional, tournament-based, etc.), crossover and mutation operators, and buggy code can generate interesting results. Some mathematicians exploit this to argue that artificial evolution is nothing more than simulated annealing, implying that there is nothing special about genetics and that it all amounts to random search and chance. I argue that they are wrong because in doing so they automatically dismiss the aspects of artificial evolution shown on the top row of figure 3. But even if they were right, they still miss the point of Evolutionary Robotics. As I said above, what matters most in this approach is to achieve self-organization of an autonomous, situated, and interactive machine. The algorithmic details of how this is achieved come in second place.

The row in the middle of figure 3 shows some of the current and most important aspects of Evolutionary Robotics. The first aspect concerns the Level at which artificial evolution operates. It can be applied to simulated organisms or to physical robots, or to a combinations of both. There is an important ongoing discussion about these issues and several strategies have been suggested to allow transfers across levels (e.g., see [22, 29]) that deserve further efforts. Similarly, one can decide to evolve the control system or some characteristics of the robot body (morphology, sensors, etc.), or co-evolve them both [25]. Finally, one may decide to physically evolve the hardware, such as the electronic circuits [36] and the body shape [33]. All these issues, among others, are likely to define a new engineering methodology.

Another aspect of Evolutionary Robotics concerns the evolutionary Mode. Should one use a single robot and serially test each individual one at a time [8], or is it better to use a population of such robots sharing the same environment [41]? What are the emerging dynamics and how do they affect the results? Within a collective system, one may set up a competitive scenario or a cooperative one. It may even happen that competition and cooperation autonomously develop as an emergent phenomenon. Interactive mode instead is the situation where a robot evolves interactively with a human who manually selects the best individuals. There are only sporadic studies of interactive evolution, but this is going to be a crucial issue for applications related to human assistance and entertainment.

Incremental mode is when one attempts to carry on evolution from previously evolved populations, usually introducing some type of modification to make the system more complex. Incremental evolution is important to tackle complex problems that cannot be evolved from scratch (the bootstrap problem), but only few studies have been dedicated to this topic so far [6, 21, 10].

The top row of figure 3 displays areas of research that have been only scarcely addressed in the literature, but are likely to make significant advancement in Evolutionary Robotics. The Evaluation aspect is concerned with the development of methodologies to set up an evolutionary system, to measure, and assess its development, and to objectively compare it to other evolutionary systems. The current situation is that every one has his own fitness recipes, most describe results in terms of average and best fitness per generation, and nobody compares results with those of other people. Although some authors have attemted to devise new ways of measuring evolutionary dynamics [2, 5], more work in this direction is needed. To this end, in the next section of this paper I will suggest a method to conceive, assess, and compare fitness functions.

Figure 4: Incremental evolution across different robots. An initial population is evolved for navigation abilities on the miniature mobile robot Khepera (left). After 100 generations the population is transferred on the larger Koala robot and incrementally evolved using the same fitness function (center). The population quickly re-adapts to the new morphology and sensor characteristics, as shown by the fitness data (right) [10].

A similar story holds for the behavioral evaluation of evolved robots. Since these robots are situated system, one cannot understand their functioning by simply looking at the evolved architectures and parameters. Earlier on we mentioned that evolved robots exploit invariant features, that is subjective constant relationships between the agent and the environment. In order to understand these invariants it will be necessary for roboticians to adopt the same techniques used by psychologists and ethologists to study invariants exploited by animals and humans. There are more than 150 years of well-established psychophysical techniques ready to be adapted to the new generation of intelligent robots.

The Genetics aspect is about what goes into the artificial chromosomes and how these chromosomes are mapped into individuals. Genetic encoding and genotype-phenotype mappings are the key to the evolvability of a system. There are some interesting studies in this area that show the potentials of better understanding this aspects. For example, Harvey's work on Neutral Networks in artificial evolution [20] indicates that some type of genetic mappings are more likely to lead to useful neutral changes (i.e., changes that do not immediately affect the fitness of the individual) that may eventually provide a species with radically new and more adaptive abilities. Several authors have explored different types of encoding and mapping schemes, but none of these strategies has shown a distinctive advantage with respect to vanilla encoding styles.[3]

Finally, the Plasticity aspect refers to all those processes that contribute to adaptively shape a fully-fledged organism. Structural growth, maturation, and ontogenetic adaptation (often called learning) are largely inexplored factors that complement, improve, and modify the adaptive properties of artificial evolution. For example, it has been shown that combining evolution with generative grammars (rules that recursively unfold into other rules) can effectively produce complex patterns that sometimes resemble life-like structures [19, 23, 35]. However, it is not clear how and what growing rules should be encoded. Nolfi and colleagues have begun to tackle the issue of maturation whereby the control system of a robot adaptively develops in time while the robot interacts with the environment, showing that this results in more efficient behaviors and specialized control architectures [32]. This seems to be the only work available in this area despite the fact that according to biologists and psychologists, maturation plays a major role in the definition of the final organism. Some people have

[3]I mean that either the same abilities can be evolved with a more straightforward encoding technique or that the evidence presented is by far not conclusive.

addressed the combination of evolution and other types of adaptive mechanisms. In one of the next sections I will propose a different approach that emphasizes evolution *of* adaptation rather than evolution *and* adaptation.

Even small advancements in any of the areas displayed on the top row of figure 3 will greatly contribute towards the creation of autonomously evolving machines that display life-like properties. I suspect that in the long run artificial evolution will play a minor-but essential–role in the overall methodology, that providing a medium for the development of powerful adaptive mechanisms such as growth, development, and ontogenetic adaptation.

3 Fitness Space

The fitness function defines which individuals are selected for reproduction. It is therefore a major factor of artificial evolution. If there is no fitness function and individuals are randomly selected, the effects of reproduction amount to genetic drift whereby all individuals become similar to each other with small random variations.

In Evolutionary Robotics, the choice of fitness function has strong consequences for implementation on physical robots, evolvability of the robot, dynamics of the evolutionary process, and eventually for outcomes. Most people struggle with the choice of suitable functions using a trial-and-error strategy. The most affected by this choice are people interested in using artificial evolution for Automated Engineering because they often have well-defined expectations about the final result. Unfortunately, there is not a way to infer a fitness function from the definition of the expected behavior. Typically, one comes up with a function based on one's own experience, tries it out, and then gradually modifies it to accommodate additional constraints. Although conceiving a fitness function suitable for a desired ability is still much easier than designing the corresponding program, the widespreading use of Evolutionary Robotics requires better awareness of the decisions involved in setting up a fitness function.

In this section I propose *Fitness Space* as a framework to devise, assess, and compare fitness functions. Fitness Space can be used as a guideline to come up with fitness functions according to one's goals, but it does not provide a recipe to actually define specific functions. Fitness Space is defined by three dimensions.[4]

3.1 Functional-Behavioral Dimension

The first dimension is given by the continuum between Functional and Behavioral fitness. A purely functional fitness is based only on components that directly measure the way in which the system functions. For example, in an early attempt to evolve a neural controller for a walking robot, Lewis and Fagg used a functional fitness that measured the frequency and amplitude of the oscillations of the evolutionary controller [24]. The closer these two components were to the desired pattern, the higher the fitness of the individual.[5] This implies that the authors knew what type of oscillatory dynamics were required for producing a certain behavior. On the other end, a purely behavioral fitness is based only on components that measure the behavior of the individual's behavior. To stay with the example of the walking robot, a behavioral function would be proportional to the distance covered by the robot in a given amount of time. Another way of describing the difference between these two fitness extremes

[4]Fitness Space should not be confused with Fitness Landscape which instead describes the distribution of fitness values corresponding to all possible combinations of genetic states.

[5]Functional fitness was used mainly in an early stage of evolution. In later stages, the authors added further behavioral components to the fitness.

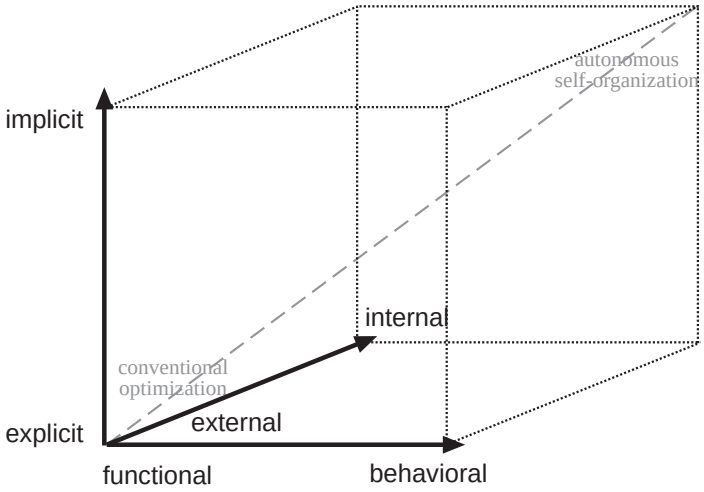

Figure 5: Fitness Space is a framework for defining and comparing fitness functions along three dimensions. It provides a nominal and ordinal scaling of functions. The diagonal between the lower-left corner and the upper-right corner defines a continuum of functions between conventional optimization approaches and generation of autonomous self-organizing systems.

is that functional fitness measures the *causes* of behavior whereas behavioral fitness measures the *effects* of behavior. Either choice has strong implications. A functional fitness can ensure the highest match between evolved and desired behavior, but it is quickly compromised as soon as mechanical or environmental factors do not match any longer the functional aspects of the controller. For example, a wheel may gradually wear out and induce a rotation bias of the robot. On the other hand, behavioral fitness can produce viable results even in the presence of some mechanical and sensory defects because the control system will implicitly accommodate them, but the results may not match the expectations. For example, the evolved robot may crawl instead of walk. The position of a given fitness function along the Functional-Behavioral dimension depends on the number of these two types of components and their relative weights. Later on we shall see in more detail how different functions can be positioned and compared in Fitness Space.

3.2 External-Internal Dimension

The dimension along the External-Internal continuum refers to availability of the fitness measure with respect to the robot. An external fitness component is one that cannot be measured directly by the robot. For example, the exact distance between a robot and an obstacle can be measured only by external positioning devices or by an external observer.[6] An internal component instead in one that can be measured by the robot itself, such as the energy level or the state of its own sensors. The difference is subtle, but very important in Evolutionary Robotics. For example, external fitness functions are often used in software simulations of robots where all aspects of the system are directly available to the programmer. Here the distinction between internal

[6]Notice that odometry and proximity sensors (such as sonar and active infrared) cannot be reliably used to estimate distances.

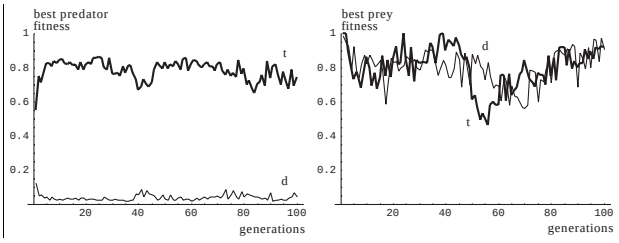

Figure 6: Comparisons between fitness of the best individuals measured as *time to contact* (`t`) and as *distance*. (`d`). Both species have been evolved using fitness `t`. **Left**: Best evolved predators do not attempt to minimize distance. **Right**: Best evolved prey attempt to maximize distance.

and external variables is only formal because both types of variables are readily available to the programmer. External fitness functions are popular because they allow a detailed and precise assessment of the robot performance and also because they naturally conform to the perspective of an external observer (in other words, it is easier to design a function that fits one's perspective of the expected behavior). However, real robots cannot always be evolved using external fitness components and, when feasible, this implies resorting to *ad hoc* and expensive devices, such as a Global Positioning System or an external camera with tracking software. Therefore, the choice of an external function should be carefully evaluated when devising an evolutionary robotic system. For example, external fitness functions are not recommended if one decides to start evolving the robot in simulation and later wishes to incrementally carry on evolution on a physical robot. Similarly, one should be careful about making strong claims on results obtained using external fitness functions because these results might not be easily generalized to many real-world situations where the necessary measuring devices are not available.

In the context of autonomous robots, we think that external fitness functions do not give a higher probability of success than internal fitness functions because they are based on the perspective of an external observer. In other words, the robot may be forced to display a behavior that is too difficult for the characteristics of their sensory-motor apparatus or of their control architecture. Consider for example the case of competitive co-evolution between predator and prey physical robots that we explored in previous work [7]. In that case we used internal fitness functions based on time-to-contact between the two robots measured through the clock of the on-board microcontrollers. The fitness of the prey was *proportional* to the amount of time spent without being touched by the predator, whereas the fitness of the predator was *inversely proportional* to the amount of time spent before catching the prey. In few generations the best individuals of each species were capable of maximizing their own time-based fitness displaying a high variety of behaviors to achieve that. Since in earlier work on simulated predator-prey Cliff and Miller employed an external fitness function based on distance between the robots (proportional for the prey and inversely proportional for the predator) [5], we repeated my experiments in simulation to check whether the robots evolved with time-based fitness reported the same values when measured with distance-based fitness [13]. It turned out that while almost all best prey reported the same fitness values using the two measures, all predator robots reported very low distance-based fitness (figure 6)! The reason was that instead of pursuing the prey, they used other strategies such as waiting for the prey by a wall like a spider or attacking only when the prey was moving in a certain direction and

relative position. This meant that most of the time they were far from the prey (low distance-based fitness value), but were still very successful at catching it (high time-based fitness). This result can be understood if one considers that the prey can go faster than the predator and therefore most strategies that attempt to get closer to the prey all the time will fail to catch it. Instead, a fitness based on time to contact does not necessarily select individuals for their ability to follow prey, giving more freedom to the evolving system. This example does not imply that internal fitness functions always mean less constraints for an evolving system, but they do encourage the engineer to reconsider assumptions deriving from an external perspective.

3.3 Explicit-Implicit Dimension

The Explicit-Implicit dimension refers to the quantity of constraints explicitly imposed by a human person to select individuals for reproduction. An approximate indicator is given by the number of components included in the fitness function. The higher the number of components, the more explicit the fitness function is.

Artificial Life approaches aimed at studying evolution of ecosystems tend to use implicit fitness functions in order to ensure ecological validity because in real life there is not an explicit fitness function. For example, artificial organisms may reproduce only if and when their energy levels reach a certain threshold (these type of ecosystems are also known as *Latent Energy Environments* [28]). Compare this with a situation where the fitness function explicitly rewards –for example– the quantity of food items gathered, distance from predators, and the ability to recognize conspecifics.

Automated Engineering approaches instead tend to resort to more explicit fitness functions in the attempt to actively steer the evolutionary system towards desired behaviors. This may sound reasonable, but in practice it gets out of control very quickly. As the number of constraints increases one is faced with the problem of how to weight and combine them (addition, product, e.g.). Furthermore, a higher number of components can increase the probability of local minima and make the problem too hard for an initial random population (bootstrap problem). Once again, these problems can be partly explained by the fact that fitness constraints are chosen from an external perspective and thus may be hard to meet by the robotic hardware and control architecture.

We think that explicit fitness functions are in contrast with the search of emergent forms of artificial intelligence because although the resulting evolved systems have not been pre-programmed their abilities have largely been decided and constrained by an external observer. Such evolved systems can hardly display unexpected abilities and under some definitions they may not be called emergent (for example, when emergence is defined as the degree of surprise [34]).

3.4 Comparing Evolutionary Experiments

Fitness Space allows us to compare the growing number of experiments available in the literature and make their underlying approach more explicit. In post-Galileian science, there are four methods to compare, or scale, experimental observations. *Categorical or nominal scales*, the most primitive methods, group observations into qualitative classes. For example, one may classify approaches in robotics as "bio-inspired", "adaptive", "cartesian", etc. *Ordinal scales* are possible only when one can assign to each observation a number that reflects some quantity property. Ordinal scale can tell only whether there is a difference between two observations and in what direction the difference goes. For example, we may say one mineral is softer than another because it is damaged when they are scratched together. However, with ordinal scales we do not

know the true magnitude of oberved phenomena. *Interval scales* are possible when we can tell precisely the difference between two obervations. For example, temperature is expressed in interval scales such as the Celsius scales. If we measure the temperatures of two objects, we can say exactly that one has –for example– 20 more units than the other. However, we cannot say that one object has double temperature than the other. *Ratio scales* allow us to say exactly that. For example, the lengths of two objects is expressed on a ratio scale. In this case we can say that an object is twice as long as the other. Interested readers will find a classic treatment of scaling methods in [38].

Fitness Space supports nominal and ordinal scales to evaluate evolutionary experiments. For example, the diagonal between the lower-left corner and the upper-right corner defines a continuum between conventional optimization approaches and self-organization of autonomous systems (figure 5). Although the point of separation between the two approaches is fuzzy, we can say that experiments falling in the lower-left region are concerned with automatic engineering whereas experiments falling in the upper-right region are concerned with emergent autonomous systems.

We can also order two experiments according to the components of their fitness functions. For example, consider two imaginary experiments aimed at evolving walking controllers for legged robots. The components used in the fitness functions are:
a=oscillation frequency of leg controller (functional, internal);
b=distance covered (behavioral, external);
c=state of motors (behavioral, internal);
d=state of bump sensor under the belly (behavioral, internal);
2=constant (affects relative position along implicit/explicit dimension).

Fitness function f^1

$$f^1 = (2 * a) + (b * d) \tag{1}$$

is composed of two additive parts. The first part rewards the controller for producing pre-determined oscillation frequencies that correspond to a desired motion pattern for each leg. This part has a strong weight (factor 2). The second part instead adds to it rewards for the distance covered by the robot body multiplied by the state of the sensors. In other words, robots that move longer without creeping over the floor get higher fitness.

Fitness function f^2

$$f^2 = c + d \tag{2}$$

has two parts too. The first part is maximized by the quantity of current sent to the motors and the second is maximized when the robot does not touch the floor. In other words, robots that keep their legs moving but do not stay on the floor will receive higher fitness.

Function f^1 therefore is *less implicit, less internal, and less behavioral* than function f^2. Assuming that all other conditions are the same, fitness f^1 can generate efficient and specific gaits (depending on the values of a), but it may take more generations and a larger population because the number of constraints are likely to make the fitness lansdcape very hard. Also, it requires the use of additional hardware to measure the distance covered by the robot. Fitness f^2 instead is not guaranteed to generate efficient gaits, in fact it may well generate robots that dance without covering much forward distance. However, the evolved solutions will be more dependent on the interactions between the robot and its environment. Also, this function does not require additional devices and is portable to a range of different robots with unknown dynamics and kinematics because it does not require functional knowledge of the system.

When comparing evolutionary robots though the fitness function is not the only factor that determines the outcomes and evolvability of a system. Although it plays

an important role, other determinant factors include the type of sensors used, the environmental setup, the type of challenge that is being addressed, and aspects of the controller architecture and genetic encoding.

4 Evolution of Adaptation

In the following sections we shall introduce a methodology to evolve robots that are capable to withstand different sources of environmental changes both during and after evolutionary selection. This is part of our effort to make artificial evolution more applicable to real-world applications without compromising on the autonomy of the robot. The method is based on evolution of mechanisms for adaptation of parameters, instead of evolution of the parameters themeselves as in the conventional approach. The fitness function is internal, behavioral, and not too explicit.

4.1 Coping with Change

The situated nature of Evolutionary Robotics is such that often evolved controllers find surprisingly simple –yet efficient– solutions that capitalize upon unexpected invariants of the interaction between the robot and its environment. For example, a robot evolved for the ability to discriminate between shapes can do so without resorting to expensive image processing techniques by simply checking the correlated activity of two receptors located in strategic positions on the retinal surface [21]. Analogously, a robot evolved for finding a hidden location can display the performances similar to those obtained by rats trained under the same conditions without resorting to complex environmental representations by using simple sensory-motor sequences that exploit geometric invariants of the environment [26]. The remarkable simplicity[7] and efficiency of these solutions is a clear advantage for fast and real-time operation required from autonomous robots, but it raises the issue of robustness when environmental conditions change. Environmental changes can be a problem also for other approaches (programming, learning, e.g.) to the extent in which the sources of change have not been considered during system design, but they are even more so for evolved systems because these often rely on environmental aspects that are often not predictable by an external observer.

Environmental changes can be induced by several factors such as modifications of the sensory appearance of objects (e.g., different light conditions), changes in sensor response, re-arrangement of environment configuration, transfer from simulated to physical robots, and transfer across different robotic platforms.

Some authors have suggested to improve the robustness of evolved systems by adding noise [29, 22] and by evaluating fitness values in several different environments [37]. However, both techniques imply that one knows in advance what makes the evolved solution brittle in the face of future changes in order to choose a suitable type of noise and of environmental variability during evolutionary training. Another approach consists of combining evolution and learning "during life" of the individual (see [3] for a comprehensive review of the combination of evolution and learning). This strategy not only can improve the search properties of artificial evolution, but can also make the controller more robust to changes that occur faster than the evolutionary time scale (i.e., changes that occur during the life of an individual) [31]. This is typically achieved by evolving neural controllers that learn with an off-the-shelf algorithm, such as reinforcement learning or back-propagation, starting from synaptic

[7] This does not imply that evolutionary approaches are restricted to forms of reactive intelligence; see for example [8]

weights specified on the genetic string of the individual [1, 32]. Only initial synaptic weights are evolved. A limitation of this approach is the "Baldwin effect", whereby the evolutionary costs associated with learning give a selective advantage to the genetic assimilation of learned properties and consequently reduce the plasticity of the system over time [27].

Here we suggest to *evolve the adaptive characteristics* of a controller instead of combining evolution with off-the-shelf algorithms. The method consists of encoding on the genotype a set of four local Hebb rules for each synapse, but *not the synaptic weights*, and let these synapse use these rules to adapt their weights online starting always from random values at the beginning of the life. Since the synaptic weights are not encoded on the genetic string, there cannot be genetic assimilation of abilities developed during life. In other words, these controller can rely less on genetically-inherited invariants and must develop on-the-fly the connection weights necessary to achieve the task. At the same time, the evolutionary cost of adaptation (i.e., the time and energy spent to adapt goes to the detriment of the individual's fitness) implicitly puts pressure for the generation of fast-adaptive architectures.

In preliminary investigations comparing evolution of genetically-determined weights with evolution of adaptive controllers on a simple navigation task, we have shown that the latter approach generates similarly-good performances in less generations [10] by taking advantage of the combined search methods. Later, we showed that evolution of adaptive controllers significantly alters the performance of robots that must cope with dynamic environments and described an experiment where co-evolutionary adaptive predators adapted on line to co-evolutionary prey robots [11].

Here we describe a new set of experiments designed to further show that this approach can generate more complex controllers and test its robustness to environmental changes that were not included during evolutionary training. For what concerns adaptation to change, here we focus on transfer of evolved controllers across different robotic platforms whose sensory-motor characteristics require partial re-configuration of the control system. In another set of forthcoming papers, we also show that this approach is effective for environmental changes that involve new sensory characteristics and new spatial relationships of the environment [40] and in transfers from simulations to physical robots without additional evolution [39].

In the next sections we give an overview of the evolutionary method and describe its application to a complex sequential task. We then present the results on the transfer of evolved across different robotic platforms. Finally, we discuss the future perspectives of this new evolutionary approach.

4.2 Encoding Mechanisms of Adaptation

The artificial chromosome encodes a set of four modification rules for each component of the neural network (components can be individual synapses or groups of synapses that converge towards the same neuron, as we shall see below), but not the synaptic strengths of the network. Whenever an artificial chromosome is decoded into a neural controller, the synaptic strengths are set to small random values. This means that the robot will initially display random actions both at generation 0 and at later generations. However, as time goes the synapses start to change their value using the genetically specified rules every 100 ms (the time necessary for a full sensory-motor loop on the physical robot). Notice that synaptic adaptation occurs on-line while the robot moves and that the network self-organizes without external supervision and reinforcement signals. The fitness function is evaluated along the whole duration of the robot "life". This introduces an implicit learning cost [27] that gives selective advantage to individuals that can adapt faster. At the end of the life, the final synaptic

strengths are not "written back" into the artificial chromosome.[8]

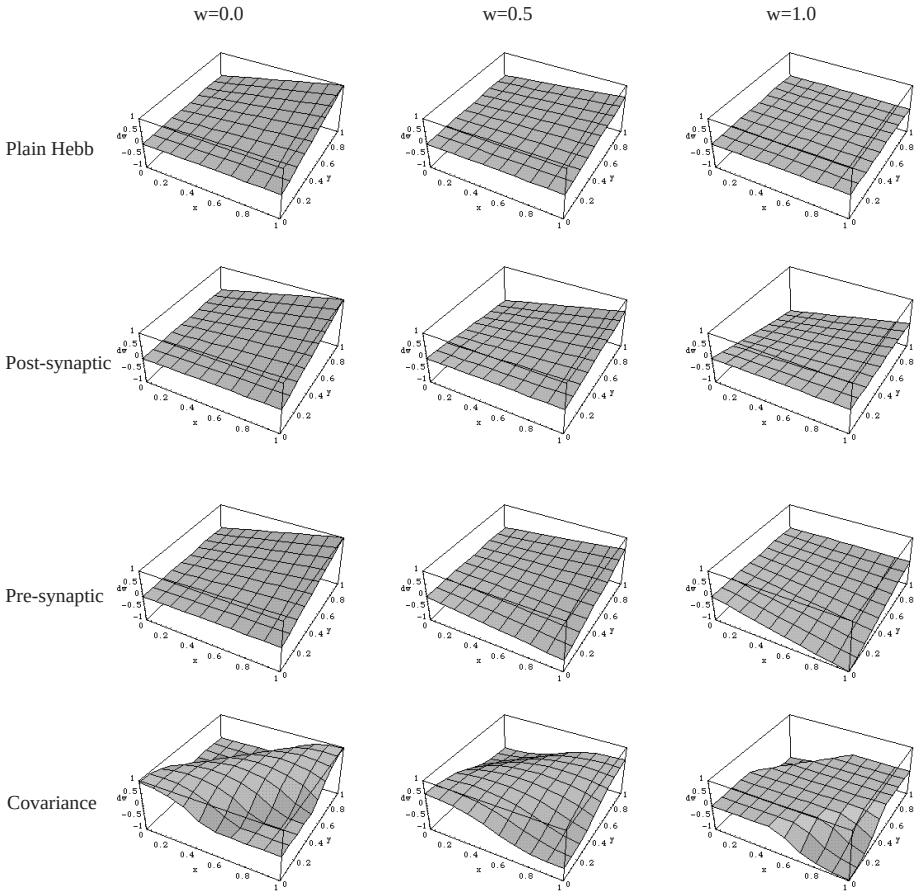

Figure 7: Synaptic change for each of the four Hebb rules. Notice that this is the *amount of change* Δw added to the synapses, not the synaptic strength. Each graph indicates the amount of change as a function of istantaneous presynaptic x and postsynaptic y activity. The amount of change also depends on the current strength w of the synapse so that synapses are always bound between 0 and 1. Three graphs are shown for each rule, in the case of current strength 0.0, 0.5, and 1.0 respectively.

We have selected four types of modification rules (figure 7) to be encoded on the artificial chromosome. The choice has been based on neurophysiological findings and on computational constraints of local adaptation. In other words, these rules capture some of the most common mechanisms of local synaptic adaptation found in the nervous systems of mammalians [42]. These rules were modified in order to satisfy the following constraints. Synaptic strength could not grow indefinitely, but was kept in the range $[0, 1]$ by means of a self-limiting mechanism which depended on synaptic strength. Because of this self-limiting factor, a synapse could not change sign, which was genetically specified, but only strength. Each synaptic weight w_{ij} is randomly

[8]In other words, we use Darwinian evolution instead of Lamarckian evolution where the effects of learning are encoded in the artificial chromosome. See [43] for an experimental comparison between these two types of evolution in changing environments.

Encoding	Bits for one synapse / node				
Genotype	1	2	3	4	5
A	sign	strength			
B	sign	Hebb rule		rate	
C	sign	strength		noise	

Table 2: Genetic encoding of synaptic parameters for Synapse Encoding and Node Encoding. In the latter case the sign encoded on the first bit is applied to all outgoing synapses whereas the properties encoded on the remaining four bits are applied to all incoming synapses. A: Genetically determined controllers; B: Adaptive synapse controllers; C: Noisy synapse controllers.

initialized at the beginning of the individual's life and can be updated after every sensory-motor cycle (100 ms),

$$w_{ij}^{t} = w_{ij}^{t-1} + \eta \Delta w_{ij},$$

where $0.0 < \eta < 1.0$ is the learning rate and Δw_{ij} is one of the four modification rules specified in the genotype:[9]

1. *Plain Hebb rule*: can only strengthen the synapse proportionally to the correlated activity of the pre- and post-synaptic neurons.

$$\Delta w = (1 - w)\, xy \tag{3}$$

2. *Postsynaptic rule*: behaves as the plain Hebb rule, but in addition it weakens the synapse when the postsynaptic node is active but the presynaptic is not.

$$\Delta w = w\,(-1 + x)\, y + (1 - w)\, xy \tag{4}$$

3. *Presynaptic rule*: weakening occurs when the presynaptic unit is active but the postsynaptic is not.

$$\Delta w = wx\,(-1 + y) + (1 - w)\, xy \tag{5}$$

4. *Covariance rule*: strengthens the synapse whenever the difference between the activations of the two neurons is less than half their maximum activity, otherwise the synapse is weakened. In other words, this rule makes the synapse stronger when the two neurons have synchronous activity.

$$\Delta w = \begin{cases} (1 - w)\mathcal{F}(x, y) & \text{if } \mathcal{F}(x, y) > 0 \\ (w)\mathcal{F}(x, y) & \text{otherwise} \end{cases} \tag{6}$$

where $\mathcal{F}(x, y) = \tanh(4(1 - |x - y|) - 2)$ is a measure of the difference between the presynaptic and postsynaptic activity. $\mathcal{F}(x, y) > 0$ if the difference is bigger or equal to 0.5 (half the maximum node activation) and $\mathcal{F}(x, y) < 0$ if the difference is smaller than 0.5.

The genes are composed of five bits. The first bit represents the sugn of the synapse. What is encoded on the remaining four bits depends on the evolutionary condition chosen (table 4.2), namely:

[9]These four rules co-exist within the same network.

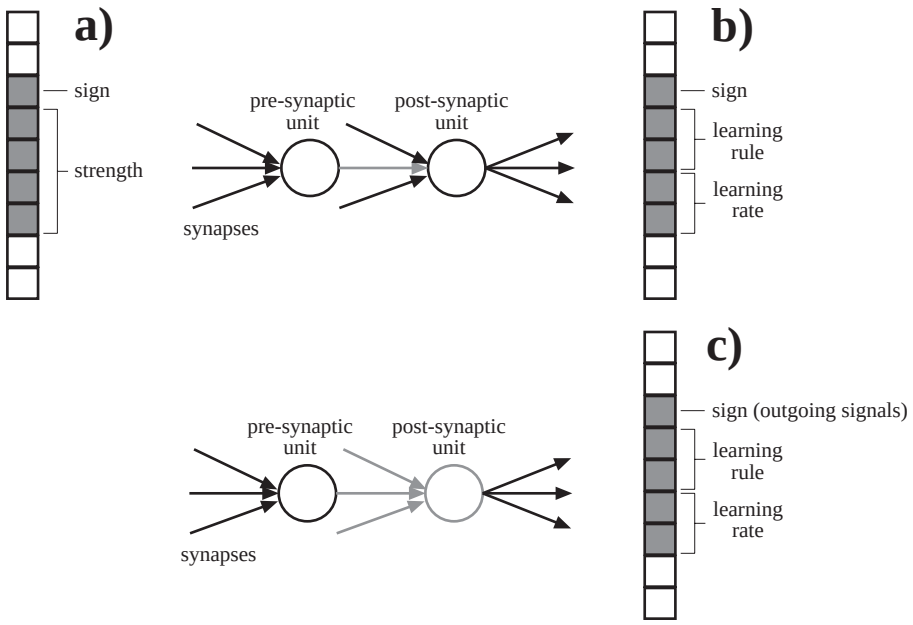

Figure 8: Different type of Genetic Encoding. **a)** Synapse Encoding for genetically-determined networks: The sign and strength of each synapse is encoded on the genotype. **b)** Synapse Encoding for adaptive networks: The sign, four learning rules, and the learning rate of each synapse is encoded on the genotype. **c)** Node Encoding for adaptive networks: all outgoing signals have the same sign and all incoming synapses have the same learning rate and learning rule. In cases **b** and **c**, the initial strength of the synapse is always set to small random values. Node Encoding cannot be applied to for genetically-determined synapses.

1. *Genetically-determined*: 4 bits encode the synaptic strength. This value is constant during "life".
2. *Adaptive synapses*: 2 bits encode 4 adaptive rules and 2 bits the learning rate. Synaptic weights are always randomly initialized at the beginning of an individual's life and then updated according to their own adaptation rule.
3. *Noisy synapses*: 2 bits encode the weight strength and 2 bits a noise range. The synaptic strength is genetically determined at birth, but a random value extracted from the noise range is freshly computed and added after each sensory motor cycle. This latter condition is used as a control condition to check whether the effects of Hebbian adaptation (condition above) are equivalent to random synaptic variability.

Two types of genetic encoding have been considered (figure 8). In the simplest case, known as *Synapse Encoding*, each synapse has its properties (one of the three described above) specified in the artificial chromosome. In *Node Encoding* the properties describe are attributes of each node: the sign bit corresponds to the sign of all the outgoing synapses while the remaining 4 bits apply to all incoming synapses for that node. Synapse Encoding allows a detailed definition of the controller, but for a fully connected network of N neurons the genetic length is proportional to N^2. Instead Node Encoding requires a much shorter genetic length (proportional to N), but it al-

Figure 9: A mobile robot Khepera equipped with a vision module gains fitness by staying on the gray area only when the light is on. The light is normally off, but it can be switched on if the robot passes over the black area positioned on the other side of the arena. The robot can detect ambient light and the color of the wall, but not the color of the floor.

lows only a rough definition of the controller. In recent work [15] we showed that our evolutionary adaptive approach does not need a lengthy direct representation because the actual weights of the synapses are always shaped at run-time by the genetically specified rules. However, this is not possible in the traditional approaches where it is necessary to assign good initial weights to the controller. Therefore, the experiments reported in this paper compare evolution of genetically-determined networks using Synapse Encoding with evolution of adaptive networks using Node Encoding.

4.3 A Sequential Task: The "Light-Switching" Problem

In this set of experiments, we have compared the performance of evolutionary adaptive controllers with respect to evolution of synaptic weights and evolution of noisy synapses in a sequential task.

A mobile robot Khepera equipped with a vision module is positioned in the rectangular environment shown in figure 9. A light bulb is attached on one side of the environment. This light is normally off, but it can be switched on when the robot passes over a black-painted area on the opposite side of the environment. A black stripe is painted on the wall over the light-switch area. Each individual of the population is tested on the same robot, one at a time, for 500 sensory motor cycles, each cycle lasting 100 ms. At the beginning of an individual's life, the robot is positioned at a random position and orientation and the light is off.

The fitness function is given by the number of sensory motor cycles spent by the robot on the gray area beneath the light bulb *when the light is on* divided by the total number of cycles available (500). In order to maximize this fitness function, the robot should find the light-switch area, go there in order to switch the light on, and then move towards the light as soon as possible, and stand on the gray area. Since this sequence of actions takes time (several sensory motor cycles), the fitness of a robot will never be 1.0. Also, a robot that cannot manage to complete the entire sequence will be scored with 0.0 fitness. A light sensor placed under the robot is used to detect the color of the floor—white, gray, or black— and passed to a host computer in order to switch on the light bulb and compute fitness values. The output of this sensor is *not* given as input to the neural controller. After 500 sensory motor cycles, the light is switched off and the robot is repositioned by applying random speeds to the wheels for 5 seconds.

Notice that the fitness function does not explicitly reward this sequence of actions (which is based on our external perspective), but only the final outcome of the sequence

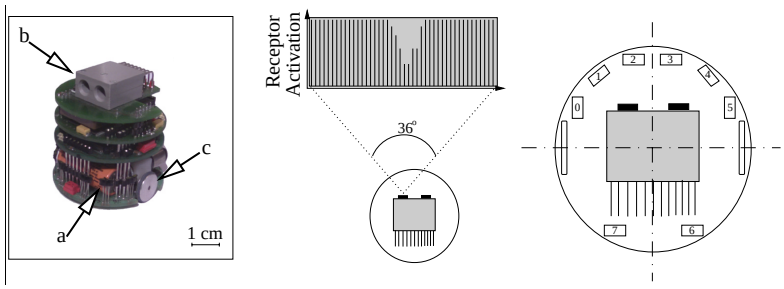

Figure 10: The Khepera robot used in the experiments. Infrared sensors (a) measure object proximity and light intensity. The linear vision module (b) is composed of 64 photoreceptors covering a visual field of 36° (center). The output of the controller generates the motor commands (c) for the robot. Right figure shows the sensory disposition of the Khepera robot.

of behaviors chosen by the robot. In Fitness Space (figure 5) this function is behavioral, internal (the computation is based on variables read through the sensors of the robot), and almost implicit (only one component is used).

The robot has The controller we have used in our experiments is a fully-recurrent discrete-time neural network. It has access to three types of sensory information from the robot (figures 10 and 11):

1. *Infrared light*: the active infrared sensors positioned around the robot (figure 10, a) measure the distance from objects. Their values are pooled into four pairs and the average reading of each pair is passed to a corresponding neuron.
2. *Ambient light*: the same sensors are used to measure ambient light too. These readings are pooled into three groups and the average values are passed to the corresponding three light neurons.
3. *Vision*: the vision module (figure 10, b) consists of an array of 64 photoreceptors covering a visual field of 36° (figure 10, center). The visual field is divided up in three sectors and the average value of the photoreceptors (256 gray levels) within each sector is passed to the corresponding vision neuron.

Two motor neurons are used to set the rotation speed of the wheels (figure 10, c). Neurons are updated every 100 ms.

The fitness results reported in figure 12 show that individuals with adaptive synapses and Node Encoding (graph on the left) are much better than individuals with genetically-determined synapses and Synapse Encoding (graph in the center) in that:

1. Both the fitness of the best individuals and of the population report higher values (0.6 against 0.5).
2. They reach the best value obtained by genetically-determined individuals in less than half generations (40 against more than 100).

Figure 13 shows the behaviors of two best individuals evolved with adaptive synapses and Node Encoding (left) and with genetically-determined weights and Synapse Encoding (right). In both cases individuals aim at the area with the light switch[10] and, once the light is turned on, they move towards the light and remain there. The better

[10]Their performance is badly affected if the vision input is disabled, indicating that they do not use random search to locate the switch (data not shown).

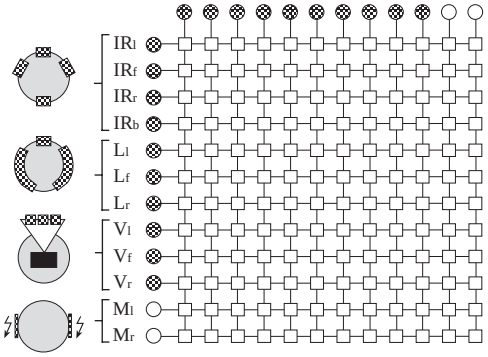

Figure 11: The neural controller is a fully-recurrent discrete-time neural network composed of 12 neurons giving a total of 12 x 12= 144 synapses (here represented as small squares of the unfolded network). 10 sensory neurons receive additional input from one corresponding pool of sensors positioned around the body of the robot shown on the left (l=left; r=right; f=front; b=back). $I\vec{R}$=Infrared Proximity sensors; $\vec{L}$=Ambient Light sensors; $\vec{V}$=vision photoreceptors. Two motor neurons $\vec{M}$ do not receive sensory input; their activation sets the speed of the wheels ($M_i > 0.5$ forward rotation; $M_i < 0.5$ backward rotation)

fitness of the adaptive controllers (given on the top of each box, see figure caption) is given by straight and faster trajectories showing a clear behavioral change between the first phase where they go towards the switching area and the second phase where they become attracted by the light. Instead, genetically-determined individuals display always the same looping trajectories around the environment with some attraction towards the stripe and the light. This minimalist behavior that depends on invariant geometrical relations of the environment gives them a chance to accomplish the task but with a lower performance.

Additional tests have been carried out to assess the role of adaptation in the behavior of the individuals with adaptive synapses. For example, one might argue that what matters is the sign of the synapse and not its strength as long as it is non-zero, or that adaptive synapses may have the same effect of fixed synapses with strengths set to their average values[11]. The results reported by our control experiments and analyses clearly indicated that evolved adaptive networks modify their parameters in ways that are functionally related to the survival criterion [15].

4.4 Cross-platform Adaptation

Cross-platform transfer is a very useful feature, but we are not aware of any control system that can be transferred across different robots without changes. Cross-platform becomes useful in adaptive and evolutionary systems where initial training experiences can cause the robot to produce harmful actions. One may train (or evolve) control systems for a desktop sturdy robot like the miniature Khepera and then download them to larger and consequently more fragile robots[12]. In this case, it would be desirable that the control system self-adapts to the new sensory-motor characteristics

[11] This latter suggestion was made by Flotzinger [16] who replicated our previous experiments on Adaptive Synapses with Synapse Encoding [9]

[12] Obviously, the two robots must share some characteristics, such as type of sensors and actuators used, that allow a suitable interfacing of the control system.

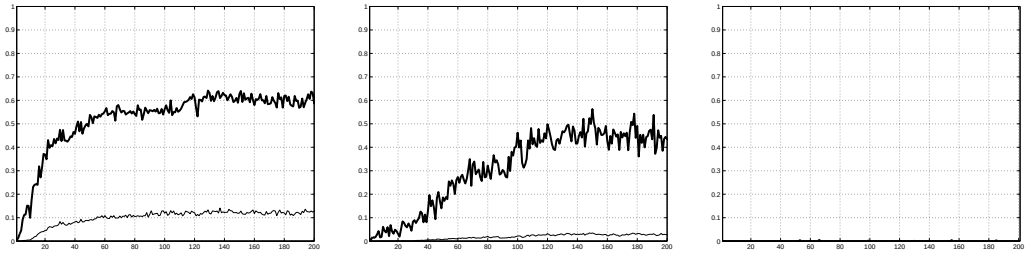

Figure 12: Comparison of adaptive synapses with Node Encoding (*left*) versus genetically-determined synapses with Synapse Encoding (*right*). Thick line=best individual; thin line=population average; dashed line=genetic diversity. Each data point is an average over 10 replications with different random initializations. Population size is 100 and 20 best individuals reproduce by making 5 copies. Crossover probability is 0.2 and mutation probability is 0.05 (per bit).

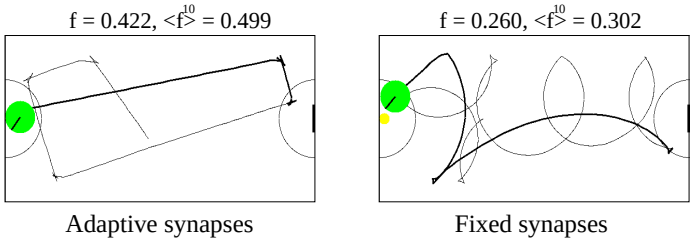

Figure 13: Behaviors of two best individuals (from last generation) with adaptive synapses and Node Encoding (*left*) and with genetically-determined synapses and Synapse Encoding (*right*). When the light is turned on, the trajectory line becomes thick. The corresponding fitness value is printed on the top of each box along with the average fitness of the same individual tested ten times from different positions and orientations.

and morphology. In previous work we have shown that this can be achieved by using incremental evolution of genetically-determined networks [10]. However, even for a simple reactive navigation behavior it took additional 20 generations to re-adapt to the new robot.

Here we test the adaptive properties of the evolutionary adaptive strategy by transferring onto a physical Koala robot (figure 14, left) the best individuals of the last generation evolved on the miniature Khepera robot. The Koala robot has six wheels driven by two motors (one on each side) and 16 infrared sensors (figure 14, right) with a different and stronger detection range.

Taking advantage of the setup offered by *TeleRoboLab*[13], a mobile robot Koala equipped with a vision module is positioned in the rectangular environment shown in figure 15. As in the previous experiment with the Khepera robot, the Koala robot must find the light-switching area, go there in order to switch the light on, and then move towards the light as soon as possible and stay there in order to score fitness points.

An external positioning system emitting laser beams at predefined angles and fre-

[13] *http://TeleRoboLab.epfl.ch* is a web site created and maintained by P. Saucy and F. Mondada that allows an external user to teleoperate a Koala robot in a physical environment.

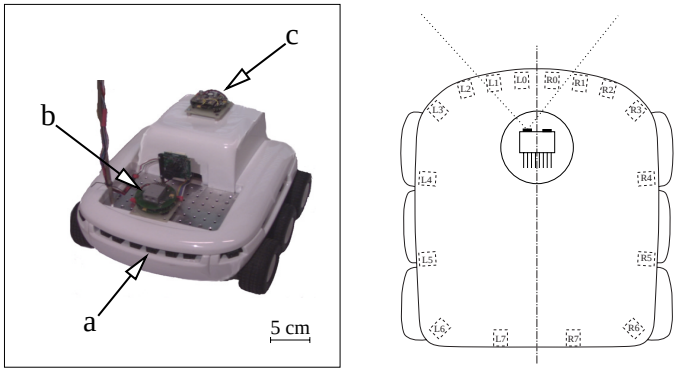

Figure 14: The Koala robot used in the experiments. Infrared sensors (a) measure object proximity and light intensity. The linear vision module (b) is the same as used in the experiments with the Khepera robot. The localization module (c) provides the position of the robot at every time step. Right figure shows the sensory layout of the Koala robot. Only 8 equally-spaced sensors are selected as input to the network.

quencies is positioned on the top of the environment and the Koala robot is equipped with an additional turret capable of detecting laser and computing in real-time the robot displacement. This information is used in order to control the light and to compute the fitness. The performance of adaptive individuals is not affected by the transfer from the Khepera robot (striped bars) to the Koala robot (dotted bars), whereas genetically-determined individuals report a significative fitness loss. Individuals with noisy synapses are not affected by the transfer because their behavior is always random and not effective in both Khepera and Koala robots (for more details see [39]).

Individuals evolved in simulation for the Khepera robot display a satisfactory behavior when tested on the Koala robot. They correctly approach the light-switching area and they are clearly attracted by light (figure 16, left). As in the case of real Khepera robot, once arrived under the light the Koala robot moves around the fitness area while remaining close to it until the testing time is over.

On the other hand, genetically-determined individuals (center) perform spiralling trajectories around the environment and do not display any attraction by the black stripe or the light. They eventually manage to pass through the light-switching area, turn the light on, and occasionally score fitness points passing through the fitness area. In several cases, genetically-determined individuals get stuck on the walls of the environment (behaviors not shown). Individuals with noisy synapses (right) score a low performance because their strategy is based in random navigation.

5 A look ahead

Over the last few years the number of projects in Evolutionary Robotics around the world has been constantly increasing. In this chapter we have described the main approaches to this field and provided a personal interpretation of more and less significant areas of research for the years to come.

We have also suggested Fitness Space as a framework to design, assess, and compare fitness functions with respect to the outcome and evolvability of evolutionary robots. Although the fitness function is not the only factor that characterize an evo-

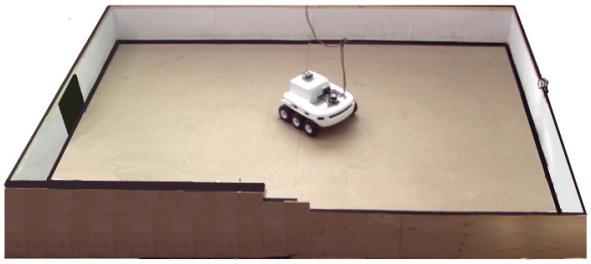

Figure 15: A mobile robot Koala equipped with a vision module gains fitness by staying near the lamp (right side) only when the light is on. The light is normally off, but it can be switched on if the robot passes near the black stripe (left side) positioned on the other side of the arena. Position of the robot is controlled by an external positioning system and passed to the computer in order to control the light and to compute the fitness.

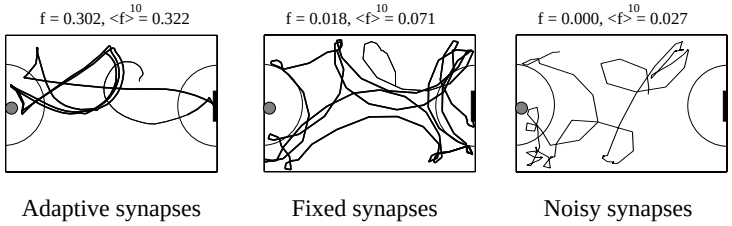

Figure 16: Behaviors of individuals with adaptive synapses (left), genetically-determined synapses (center), and noisy synapses (right) tested on the Koala robot. Individuals belong to the last generation evolved in simulation for the Khepera robot.

lutionary experiment, we think that Fitness Space is a useful tool to guide the setup of an experiment and to clarify where an experiment stands with respect to others in the literature.

The success of Evolutionary Robotics will ultimately depend by its ability to generate new robotic systems that could not be designed with conventional techniques. This means the ability to generate robots that display complex skills and can cope with unpredictable changes.

We think that the evolutionary method presented in the second part of this paper represents a significative step forward towards making Evolutionary Robotics applicable to real-world applications of autonomous robotics. In scenarios like those –for example– of robots probing an asteroid surface or robots interacting with an handicapped person it is impossible to evolve the control system on the spot (not even incrementally). However, one might reproduce the working conditions in the laboratory to some degree of approximation and evolve the adaptive controller in there. The controller would then be transferred on the final robot and let free to adapt to actual working conditions in a few seconds.

We also think that our adaptive strategy will be useful for evolving more complex and powerful control architectures. In current methods there is a trade-off between the complexity the genotype/phenotype mapping and the evolvability of such systems which is partly due to the fact that the phenotype largely depends on genetic instructions. By evolving the adaptive characteristics along with other high-level parameters

(position and type of nodes, e.g.) of the controller, one may obtain simpler genetic encodings and a higher tolerance to mutations. This would make the evolved controllers more viable, add neutrality to the genetic landscape, and ultimately improve evolvability.

6 Acknowledgements

The second part this manuscript (section 4) is based on a more extended report that will appear elsewhere [39]. Joseba Urzelai is supported by grant nr. BF197.136-AK from the Basque government. Thanks to Patrick Saucy for his help on the cross-platform experiment.

References

[1] D. H. Ackley and M. L. Littman. Interactions between learning and evolution. In C.G. Langton, J.D. Farmer, S. Rasmussen, and C. Taylor, editors, *Artificial Life II: Proceedings Volume of Santa Fe Conference*, volume XI. Addison Wesley: series of the Santa Fe Institute Studies in the Sciences of Complexities, Redwood City, CA, 1992.

[2] M. A. Bedau, E. Snyder, C. T. Brown, and N. H. Packard. A comparison of evolutionary activity in artificial evolving systems and in the biosphere. In P. Husbands and I. Harvey, editors, *Proceedings of the 4th European Conference on Artificial Life*, Cambridge, MA, 1997. MIT Press.

[3] R. K Belew and M. Mitchell, editors. *Adaptive Individuals in Evolving Populations: Models and Algorithms*. Addison-Wesley, Redwood City, CA, 1996.

[4] V. Braitenberg. *Vehicles. Experiments in Synthetic Psychology*. MIT Press, Cambridge, MA, 1984.

[5] D. Cliff and G. F. Miller. Tracking the Red Queen: Measurements of adaptive progress in co-evolutionary simulations. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, editors, *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life*, pages 200–218. Springer Verlag, Berlin, 1995.

[6] D. Floreano. Emergence of Home-Based Foraging Strategies in Ecosystems of Neural Networks. In J. Meyer, H. L. Roitblat, and S. W. Wilson, editors, *From Animals to Animats II: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. MIT Press-Bradford Books, Cambridge, MA, 1993.

[7] D. Floreano. Evolutionary Robotics in Artificial Life and Behavior Engineering. In T. Gomi, editor, *Evolutionary Robotics. From Intelligent Robots to Artificial Life*. AAI Books, Ontario, Canada, 1998.

[8] D. Floreano and F. Mondada. Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26:396–407, 1996.

[9] D. Floreano and F. Mondada. Evolution of plastic neurocontrollers for situated agents. In P. Maes, M. Matarić, J-A. Meyer, J. Pollack, H. Roitblat, and S. Wilson, editors, *From Animals to Animats IV: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, pages 402–410. MIT Press-Bradford Books, Cambridge, MA, 1996.

[10] D. Floreano and F. Mondada. Evolutionary Neurocontrollers for Autonomous Mobile Robots. *Neural Networks*, 11:1461–1478, 1998.

[11] D. Floreano and S. Nolfi. Adaptive behavior in competing co-evolving species. In P. Husbands and I. Harvey, editors, *Proceedings of the 4th European Conference on Artificial Life*, Cambridge, MA, 1997. MIT Press.

[12] D. Floreano and S. Nolfi. God Save the Red Queen! Competition in Co-evolutionary Robotics. In J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. L. Riolo, editors, *Proceedings of the 2nd International Conference on Genetic Programming*, San Mateo, CA, 1997. Morgan Kaufmann.

[13] D. Floreano, S. Nolfi, and F. Mondada. Competitive Co-Evolutionary Robotics: From Theory to Practice. In R. Pfeifer, B. Blumberg, J-A. Meyer, and S. Wilson, editors, *From Animals to Animats V: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*. MIT Press-Bradford Books, Cambridge, MA, 1998.

[14] D. Floreano, S. Nolfi, and F. Mondada. Co-evolution and ontogenetic change in competing robots. *Robotics and Autonomous Systems*, in the press, 1999.

[15] D. Floreano and J. Urzelai. Evolution of Neural Controllers with Adaptive Synapses and Compact Genetic Encoding. In D. Floreano, J-D. Nicoud, and F. Mondada, editors, *Advances in Artificial Life*. Springer Verlag, Berlin, 1999.

[16] D. Flotzinger. Evolving plastic neural network controllers for autonomous robots. Msc dissertation 9580131, COGS, University of Sussex at Brighton, 1996.

[17] C. R. Gallistel, editor. *The Organization of Learning*. MIT Press, Cambridge, MA, 1990.

[18] J. J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, 1979.

[19] F. Gruau. Automatic definition of modular neural networks. *Adaptive Behavior*, 3:151–183, 1994.

[20] I. Harvey. Artificial evolution for real problems. In T. Gomi, editor, *Evolutionary Robotics*, pages 187–220. AAI Books, Ontario, Canada, 1997.

[21] I. Harvey, P. Husbands, and D. Cliff. Seeing The Light: Artificial Evolution, Real Vision. In D. Cliff, P. Husbands, J. Meyer, and S. W. Wilson, editors, *From Animals to Animats III: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*. MIT Press-Bradford Books, Cambridge, MA, 1994.

[22] N. Jakobi. Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics. In P. Husbands and I. Harvey, editors, *Proceedings of the 4th European Conference on Artificial Life*, Cambridge, MA, 1997. MIT Press.

[23] M. Komosinski and Sz. Ulatowski. Framsticks: Towards a simulation of a nature-like world, creatures and evolution. In D. Floreano, J-D. Nicoud, and F. Mondada, editors, *Advances in Artificial Life - ECAL99*, Berlin, 1999. Springer Verlag.

[24] M. A. Lewis, A. H. Fagg, and A. Solidum. Genetic programming approach to the construction of a neural network for a walking robot. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2618–2623. IEEE Press, 1996.

[25] H. H. Lund, J. Hallam, and W-P. Lee. Evolving robot morphology. In *Proceedings of the IEEE 4th International Conference on Evolutionary Computation*. IEEE Press, 1997.

[26] H. H. Lund and O. Miglino. Evolving and Breeding Robots. In P. Husbands and J-A. Meyer, editors, *Proceedings of the First European Workshop on Evolutionary Robotics*. Springer Verlag, 1998.

[27] G. Mayley. Landscapes, Learning Costs and Genetic Assimilation. *Evolutionary Computation*, 4(3):213–234, 1996.

[28] F. Menczer and R. K. Belew. Latent energy environments. In R. K. Belew and S. Mitchell, editors, *Plastic Individuals in Evolving Populations*. Addison Wesley, Redwood City, CA, 1993.

[29] O. Miglino, H. H. Lund, and S. Nolfi. Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life*, 2:417–434, 1996.

[30] F. Mondada and D. Floreano. Evolution of neural control structures: some experiments on mobile robots. *Robotics and Autonomous Systems*, 16:183–195, 1995.

[31] S. Nolfi and D. Floreano. Learning and evolution. *Autonomous Robots*, 7(1):in the press, 1999.

[32] S. Nolfi, O. Miglino, and D. Parisi. Phenotipic Plasticity in Evolving Neural Networks. In J-D. Nicoud and P. Gaussier, editors, *Proceedings of the conference From Perception to Action*. IEEE Computer Press, Los Alamitos, CA, 1994.

[33] J. B. Pollack, H. Lipson, S. Ficici, P. Funes, G. Hornby, and R. Watson. Evolutionary techniques in physical robotics. In A. Thompson, J. Miller, T. C. Fogarty, and P. Thomson, editors, *Proceedings of the Third International Conference on Evolvable Systems: from Biology to Hardware*, Berlin, 2000. Springer Verlag.

[34] E. Ronald, M. Sipper, and M. S. Capcarrère. Design, Observation, Surprise! A Test of Emergence. *Artificial Life*, 5(3):225–240, 1999.

[35] K. Sims. Evolving 3D Morphology and Behavior by Competition. In R. Brooks and P. Maes, editors, *Proceedings of the Fourth Workshop on Artificial Life*, pages 28–39, Boston, MA, 1994. MIT Press.

[36] A. Thompson. Evolving electronic robot controllers that exploit hardware resources. In F. *et al.* Moran, editor, *Advances in Artificial Life: Proc. of ECAL95*. Springer Verlag, Barcelona, 1995.

[37] A. Thompson. On the automatic design of robust electronics through artificial evolution. In D. Mange M. Sipper and A. Prez-Uribe, editors, *Proceedings of the 2nd International Conference on Evolvable Systems: From biology to hardware (ICES98)*, pages 13–24. Springer-Verlag, Berlin, 1998.

[38] W. Torgerson. *Theory and methods of scaling*. Wiley, New York, 1958.

[39] J. Urzelai and D. Floreano. Evolutionary Robotics: Coping with Environmental Change. In J. et. al. Koza, editor, *Second International Conference on Genetic and Evolutionary Computation*, San Mateo, CA, 2000. Morgan Kaufmann.

[40] J. Urzelai and D. Floreano. Evolutionary robots with fast adaptive behavior in new environments. In T. C. Fogarty, J. Miller, A. Thompson, and P. Thomson, editors, *Third International Conference on Evolvable Systems: From Biology to Hardware (ICES2000)*, Berlin, 2000. Springer-Verlag.

[41] R. Watson, S. Ficici, and J. B. Pollack. Embodied evolution: Embodying an evolutionary algorithm in a population of robots. In P. Angeline, M. Michaelewicz, G. Schonauer, X. Yao, and Z. Zalzala, editors, *Proceedings of the 1999 Congress on Evolutionary Computation*. IEEE Press, 1999.

[42] D. Willshaw and P. Dayan. Optimal plasticity from matrix memories: What goes up must come down. *Neural Computation*, 2:85–93, 1990.

[43] Y. Yamamoto, T. Sasaki, and M. Tokoro. Adaptability of Darwinian and Lamarckian Populations toward an Unknown New World. In D. Floreano, J-D. Nicoud, and F. Mondada, editors, *Advances in Artificial Life*. Springer Verlag, Berlin, 1999.